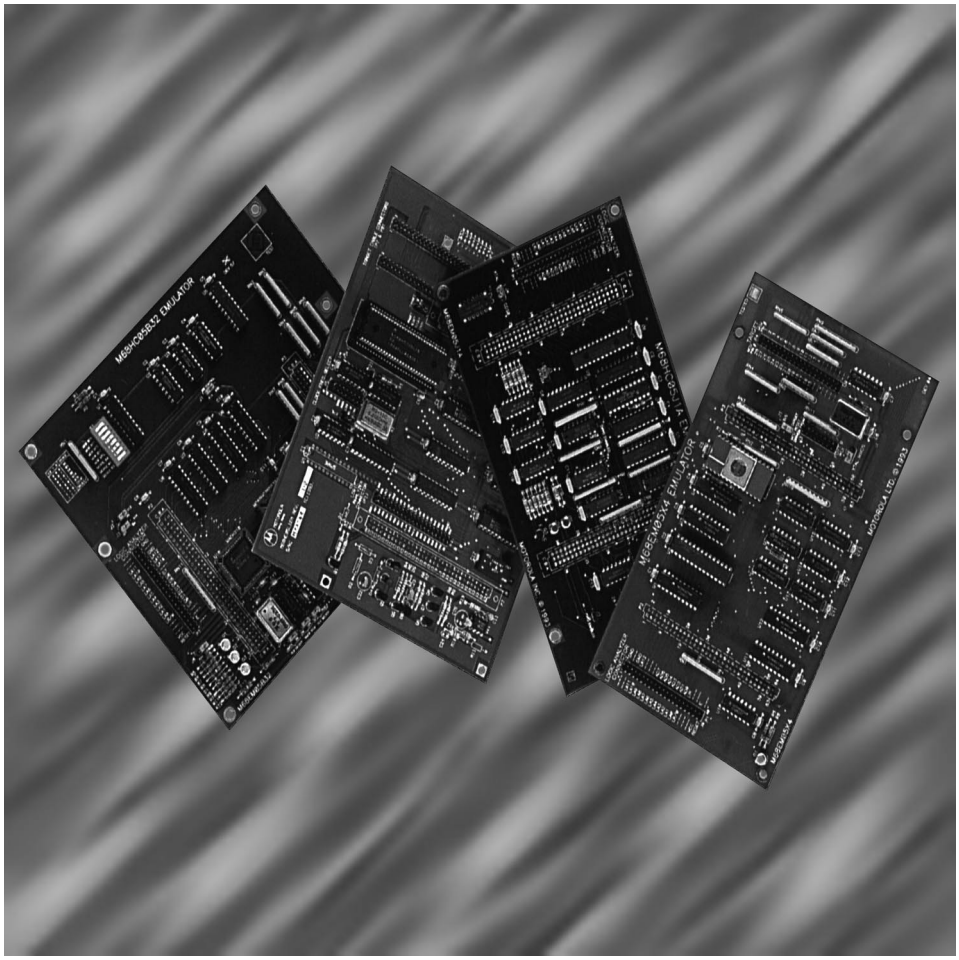
MC68EVB912DP256UM/D Rev 1
December 2000

M68EVB912DP256

EVALUATION BOARD
USER'S MANUAL



MOTOROLA

Important Notice to Users

While every effort has been made to ensure the accuracy of all information in this document, Motorola assumes no liability to any party for any loss or damage caused by errors or omissions or by statements of any kind in this document, its updates, supplements, or special editions, whether such errors are omissions or statements resulting from negligence, accident, or any other cause. Motorola further assumes no liability arising out of the application or use of any information, product, or system described herein: nor any liability for incidental or consequential damages arising from the use of this document. Motorola disclaims all warranties regarding the information contained herein, whether expressed, implied, or statutory, *including implied warranties of merchantability or fitness for a particular purpose*. Motorola makes no representation that the interconnection of products in the manner described herein will not infringe on existing or future patent rights, nor do the descriptions contained herein imply the granting or license to make, use or sell equipment constructed in accordance with this description.

Trademarks

This document includes these trademarks:

Motorola and the Motorola logo are registered trademarks of Motorola, Inc.

MCUez is a trademark of Motorola, Inc.

Apple, Macintosh, MacTerminal, and System 7 are registered trademarks of Apple Computer, Inc.

Windows and Windows 95 are registered trademarks of Microsoft Corporation in the U.S. and other countries.

Intel is a registered trademark of Intel Corporation.

Sun is a registered trademark of Sun Microsystems, Inc., in the United States and other countries.

Motorola, Inc., is an Equal Opportunity / Affirmative Action Employer.

Contents

Section 1. General Information

Section 2. Configuration and Setup

Section 3. Operation

Section 4. Hardware Reference

Appendix A. S-Record Format

Figures

Figures

Tables

Tables

Section 1. General Information

1.1 Introduction

This user's manual provides the necessary information for using the M68EVB912DP256 evaluation board (EVB), an evaluation, debugging, and code-generation tool for the MC9S12DP256 microcontroller unit (MCU).

Reference items, such as schematic diagrams and parts lists, are shipped as part of the EVB package.

1.2 General Description and Features

The EVB is an economical tool for designing and debugging code for and evaluating the operation of the MC9S12DP256 MCU. By providing the essential MCU support and input/output (I/O) circuitry, the EVB simplifies user evaluation of prototype hardware and software.

The board consists of a 9.0-inch by 4.5-inch (22.9-cm by 11.4-cm) four-layer printed circuit board (PCB) that provides the platform for interface and power connections to the MC9S12DP256 MCU chip.

Figure 1-1 shows the EVB's layout and locations of the major components, as viewed from the component side of the board.

Hardware features of the low-cost EVB include:

- Four-layer PCB
- Single-supply +12 Vdc power input
- RS-232C interface
- BDM (background debug mode) in and BDM out connectors for remote debugging of a user's target system
- Header footprints for access to all MCU pins

- 16-MHz resonator for 8-MHz bus operation
- Headers for jumper selection of and connection to hardware options (for full details of the jumper settings, refer to **Table 4-1**):
  - RS-232 selection (J6, J7, J10, J12, and J13)
  - EVB mode selection (J20, J24)
  - MCU mode selection (J33, J34, and J25)
  - Power Input (J23)
  - BDM in (J19)
  - BDM out (J22)
  - Low-voltage inhibit (LVI) reset (J14)
  - EXTAL source and access (J26)
- Twelve 2-row x 14-pin header connectors for access to the MCU's I/O and bus lines (H1through H4 and JP1 through JP8)
- Prototype expansion area for customized interfacing with the MCU
- Low-profile reset push-button switch (S1)
- LVI protection (U5)

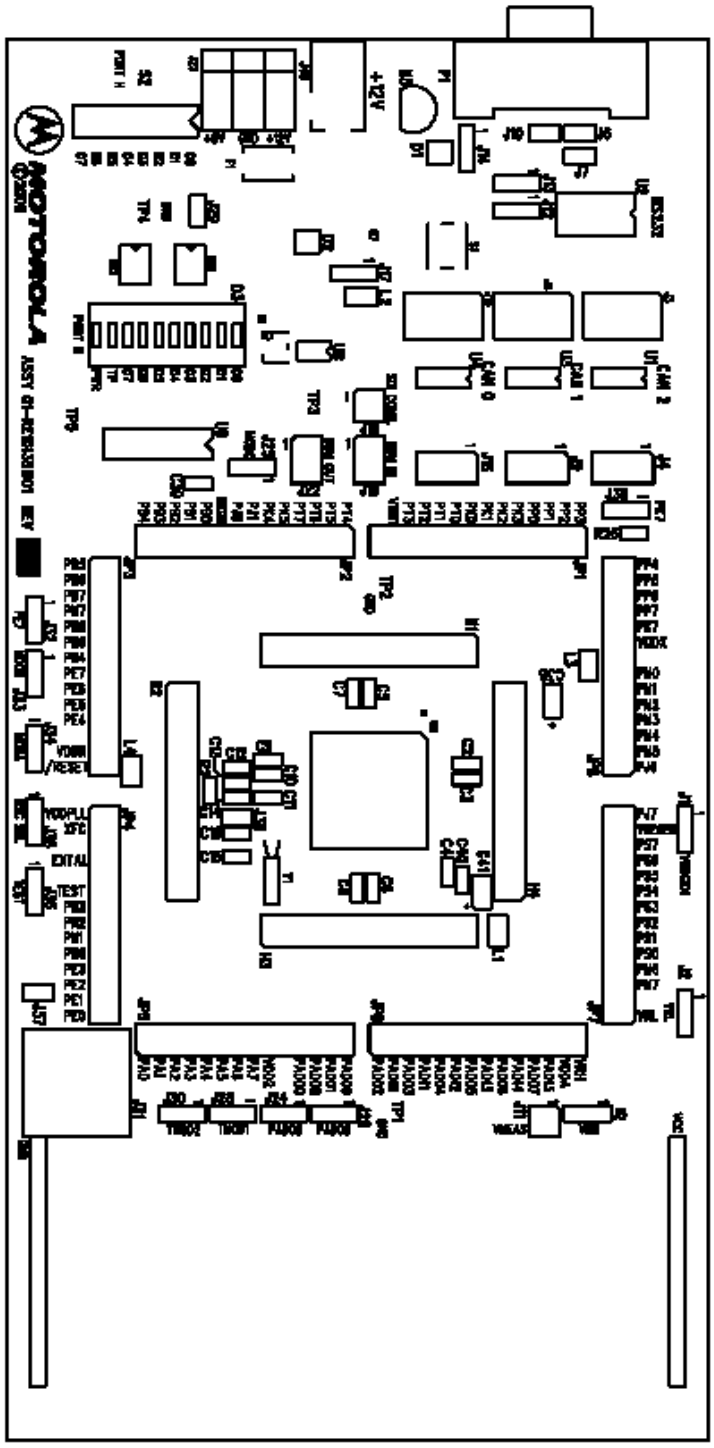


Figure 1-1. EVB Layout and Component Placement

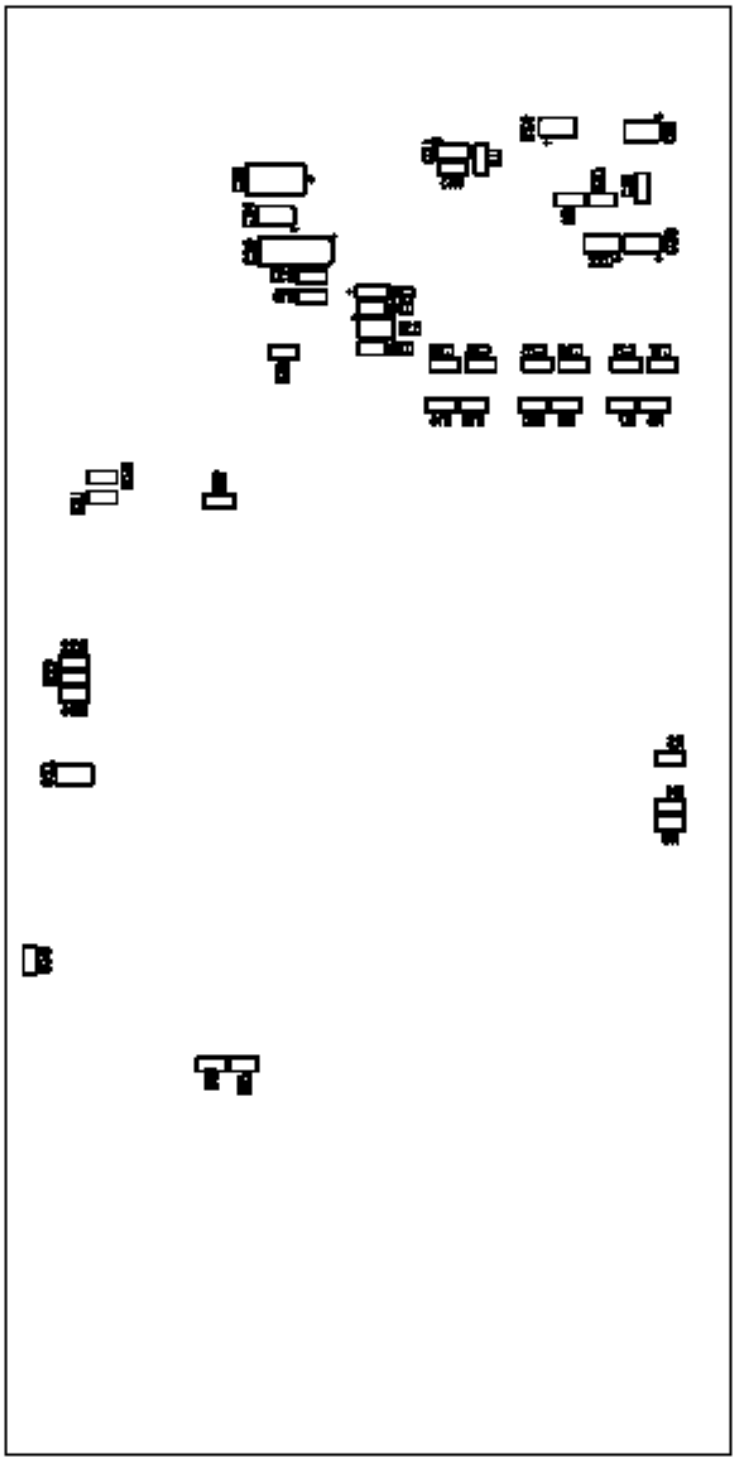


Figure 1-2. EVB Solder Side View

1.3 Functional Overview

The EVB can be configured to:

- Run a program directly out of EEPROM
- Reprogram EEPROM on the host EVB

For the correct jumper settings, refer to **4.2 Configuration Headers and Jumper Settings**.

NOTE: *EEPROM resides in two areas of memory (refer to **Table 3-1. Factory-Configuration Memory Map**), which are referred to in this manual as byte-erasable EEPROM and FLASH EEPROM. This distinction is necessary because of the different ways in which they may be programmed and used.*

If the MCU's single-wire background debug mode (BDM) interface serves as the user interface, the SCI port becomes available for user applications. This mode requires either:

- Another EVB (such as M68EVB912B32) and a host computer
- A background debug development tool, such as Motorola's serial debug interface (SDI)

For more information, refer to the *SDI™ Interface User's Manual*, Motorola document order number SDIUM/D.

Two methods may be used to generate EVB user code:

- For small programs or subroutines, the BDM interface may be used to modify memory and place object code directly into the EVB's RAM or EEPROM.
- For larger programs, HiWare's Panta Tools, P&E Microcomputer Systems' IASM12 or Motorola's MCUasm assembler may be used on a host computer to generate S-record object files, which then can be loaded into the EVB's memory using the BDM interface.

The EVB features a prototype area, which allows custom interfacing with the MCU's I/O and bus lines. These connections are broken out via headers H1through H4, which are immediately adjacent to the MCU on the board, and JP1 through JP8, which surround jumpers H1 through H4. Wire-wrap pins are

placed in these headers to connect to the prototyping area, as shown in **Figure 1-1**.

An on-board push-button switch, S1, provides for resetting the EVB hardware and restarting any code stored in the MCU's flash memory.

When operating in EVB mode, the MCU must manage the EVB hardware and serve as the user-application processor. There are a few restrictions on its use. For more information, refer to **3.7 Operational Limitations**.

1.4 External Equipment Requirements

In addition to the EVB, the following user-supplied external equipment is required:

- Power supply — See **Table 1-1** for voltage and current requirements.
- User terminal — Options:
  - RS-232C dumb terminal — may be used by user-code
  - Host computer using the MCU's BDM interface — Frees the target MCU's SCI port for user applications. This requires another EVB for use as the target or a background debug development tool, such as the Motorola serial debug interface (SDI).
- Power-supply and terminal interconnection cables as required

For full details of equipment setup, cabling, and special requirements, refer to **Section 2. Configuration and Setup**.

1.5 EVB Specifications

Table 1-1 lists the EVB specifications.

Table 1-1. EVB Specifications

Characteristic	Specifications
MCU	MC9S12DP256
MCU I/O ports	HCMOS compatible
BDM (in and out)	2-row x 3-pin headers
Communications port	RS-232C DCE port
Power requirements: 16-MHz clock source	 +12 Vdc @ 200 mA (max.) For low-voltage operation, refer to **4.8 Low-Voltage Inhibit (LVI)**
Prototype area: Area Holes	 20 x 42 Approximately 840 Approximately
Board dimensions	9.0 inches x 4.5 inches (22.9 cm x 11.4 cm)

1.6 Customer Support

To obtain information about technical support or ordering parts, call the Motorola help desk at 800-521-6274.

General Information

Section 2. Configuration and Setup

2.1 Unpacking and Preparation

Before beginning configuration and setup of the EVB:

1. Verify that these items are present in the EVB package:
   - M68EVB912DP256 board assembly
   - Documentation CD ROM
   - Warranty and registration cards
   - EVB schematic diagram and parts list
   - Assembly language development toolset
   - PANTA tools CD ROM
   - P&E Microcomputer Systems' IASM12 assembler and user's manual on CD ROM (optional)
2. Remove the EVB from its anti-static shipping bag.
3. Save all packing materials for storing and shipping the EVB.

2.2 EVB Configuration

Because the EVB has been factory-configured it is not necessary to change any of the jumper settings to begin operating immediately.

Other jumper settings affect the hardware setup and/or MCU operational modes. For an overview of all jumper-selectable functions, refer to **1.2 General Description and Features**. For details of the settings, see **Table 4-1. Jumper-Selectable Functions**.

2.3 EVB to Power Supply Connection

The EVB requires the provided power supply. See **Table 1-1. EVB Specifications** for the voltage and current specifications. For full details of the EVB's power-input circuitry, refer to **4.3 Power Input Circuitry**.

If another power supply is used, it should have current-limiting capability. If this feature is available on the power supply, set it at 200 mA. The power supply should use a 2.1 mm ID, 2.5mm OD, 11mm barrel length, and a center-positive power plug.

Optionally, the +12V DC power may be supplied through J23 terminals +12V and GND. Under normal operating conditions, the +5V terminal should not be used. To connect an external power supply through J23, use 20 AWG or smaller insulated wire. Strip each wire's insulation 1/4 inch from the end, lift the J23 contact lever to release tension on the contact, insert the bare end of the wire into J23, and close the lever to secure the wire. Observe the polarity carefully.

CAUTION: *Do not use wire larger than 20 AWG in connector P1. Larger wire could damage the connector.*

2.4 EVB to Host Debug Connection

The MCU's background debug mode (BDM in, W12) interface serves as the user interface. This setup makes the SCI port available for user applications. Additional hardware and software are required. For more information, refer to the documentation for the background debug development tool being used. This can be another EVB or a tool such as Motorola's serial debug interface (SDI).

2.5 EVB to Terminal Connection

For user-code that uses the RS-232 port, connect the terminal to P1 on the EVB, as shown in **Table 2-1**. This setup uses the MCU's SCI port and its associated RS-232C interface for communications with the terminal device.

Standard, commercially available cables may be used in most cases. Note that the EVB uses only three of the RS-232C signals. **Table 2-1** lists these signals and their pin assignments. Other signals have been routed through the RS-232C

interface chip for proper levels. Some terminal interface programs require proper levels on all pins to function correctly.

The EVB's RS-232C connector, P1, is wired as data circuit-terminating equipment (DCE) and employs a 9-pin subminiature D (DB-9) receptacle.

Most terminal devices — whether dumb terminals or the serial ports on host computers — are wired as data terminal equipment (DTE) and employ 9- or 25-pin subminiature D (DB-9 or DB-25) plugs. In these cases, normal straight-through cabling is used between the EVB and the terminal. Adapters are readily available for connecting 9-pin cables to 25-pin terminal connectors.

If the terminal device is wired as DCE, the RXD and TXD lines must be cross-connected, as shown in **Table 2-1**. Commercial null modem adapter cables are available for this purpose.

Table 2-1. RS-232C Interface Cabling

EVB P1	DTE Signal	Terminal Pins			
DCE Receptacle		DTE[1] Plug		DCE[2] Receptacle	
		DB-9	DB-25	DB-9	DB-25
2	Receive data (RXD)	2	2	3	3
3	Transmit data (TXD)	3	3	2	2
5	Ground (GND)	5	7	5	7

[1] Normal (DCE-to-DTE) cable connections
[2] Null modem (DCE-to-DCE) cable connections

Configuration and Setup

Section 3. Operation

3.1 Operating Mode

The EVB operates only in EVB mode. In the EVB mode, user code in FLASH EEPROM executes and the BDM is enabled.

In the default EVB mode, the user code in FLASH EEPROM begins execution immediately.

3.2 Startup

To operate the M68EVB912DP256, follow the startup procedure described here.

3.2.1 Startup Procedure

This startup procedure includes a checklist of configuration and setup items from **Section 2. Configuration and Setup**. To begin operating the M68EVB912DP256, follow these steps:

1. Connect the EVB to the external power supply. See **2.3 EVB to Power Supply Connection**.
2. Connect the EVB to the SDI or equivalent. See **2.4 EVB to Host Debug Connection**.
3. Apply power to the EVB and the host and perform the following steps:
   a. Verify that the host has booted correctly.
   b. Start the PANTA tools software included on the PANTA tools CD ROM. See the help files on the PANTA tools CD for more detailed instructions.
4. Reset the EVB by pressing and releasing the on-board reset switch (S1).

3.2.2 Operating Procedure

After starting the EVB in accordance with **3.2.1 Startup Procedure**, follow the operating procedure for the EVB mode

In EVB mode, the M68EVB912DP256 begins to execute code at the address contained in the reset vector at $F7FE. The code pointed to by the alternate reset vector may be a user's program in FLASH EEPROM.

User boot program — On reset, the user program executes immediately. Terminal communications take place either via the BDM interface and a serial debug interface tool such as Motorola's SDI.

3.3 Reset

EVB operation can be restarted at any time by activating the hardware reset function. To activate the hardware reset function, press and release the on-board reset switch, S1 (always applicable).

Note that the EVB's reset circuitry is associated with the low-voltage inhibit (LVI) protection. For more information, refer to **4.7 Reset** and **4.8 Low-Voltage Inhibit (LVI)**.

3.4 Aborting a User Program

When operating in EVB mode, the only way to recover from an erroneous or runaway user program is to press the reset switch (S1).

3.5 Off-Board Code Generation

Code developed outside the EVB environment should be generated with an M68HC12-compatible assembler or C compiler that can generate object files in S-record format.

S-records are described in **Appendix A. S-Record Format**.

When the S-record file has been generated, it may be loaded from the host computer into the host EVB's byte-erasable EEPROM or RAM when the host EVB is in EVB mode

More information on the EVB operating mode can be found in **3.1 Operating Mode**.

3.6 Memory Usage

The EVB's memory usage and requirements are described here and are summarized in **Table 3-1**.

3.6.1 Description

To use the FLASH EEPROM area for custom programs, refer to the MCU specification included on the documentation CD ROM.

3.6.2 Memory Map

The information in **Table 3-1** describes address ranges and locations.

Table 3-1. Factory-Configuration Memory Map

Address Range	Usage	Description
$0000 – $03FF	CPU registers	On-chip registers
$1000 – $3FFF	User code/data	12-Kbytes on-chip RAM
$0400 – $0FFF	User code/data	3-Kbytes on-chip EEPROM
$4000 – $FFBF $FFC0 – $FFFF	User-accessible functions Reset and interrupt vectors	256 Kbytes on-chip FLASH EEPROM (accessible through paging mechanism)

3.7 Operational Limitations

In EVB mode, the EVB cannot provide true emulation of a target system. These limitations are described in the following subsections.

If target-system emulation is required, the EVB may be reprogrammed and controlled via the BDM interface.

3.7.1 SCI Port Usage

The SCI port is either connected to the RS-232C RXD and TXD signals for SCI0 or SCI1 by means of jumpers J12 and J13.

3.7.2 Dedicated MCU Pins

On the EVB, the following MCU lines may be used to perform specific functions in the future. If an application requires their use, the EVB hardware and/or operating software must be custom-configured or special precautions must be taken in the application code to avoid conflicts.

PAD00 — EVB mode select pin (J24)

PAD08 — EVB mode select pin (J20)

MODA (J34) and MODB (J33) — Set MCU chip mode, normally single chip

3.7.3 Terminal Communications

High baud rates occasionally result in dropped characters on the terminal display. This is not the result of a baud rate mismatch, but is due to the host processor being too busy or too slow to process incoming data at the selected baud rate. Sometimes the problem can be ignored without harm.

If it requires correcting, try:

- Using a slower baud rate
- A different communications program
- Closing unnecessary applications or exiting Windows. In multitasking environments such as Windows® and the Macintosh System 7®, the problem can occur when several applications are running at once.
- Displaying fewer address locations or tracing fewer instructions at a time.

Operation

Section 4. Hardware Reference

4.1 Printed Circuit Board (PCB) Description

The EVB printed circuit board (PCB) is a 9.0-inch by 4.5-inch (22.9-cm by11.4-cm) board with four layers.

Most of the connection points on the EVB use headers spaced on 1/10-inch (2.54-mm) centers, with these exceptions:

- Subminiature D connector for the RS-232C interface
- External power-supply connections

4.2 Configuration Headers and Jumper Settings

For maximum flexibility, the EVB uses two types of jumper headers:

- Factory-installed headers are those most likely to be used for configuration without major alteration of the EVB's hardware operation. These headers are populated, and the factory-installed jumpers on them are preset for the default EVB hardware and firmware configurations. **Table 4-1** lists these headers by function and describes their default and optional jumper settings.
- Cut-trace header footprints offer EVB hardware options that are less likely to be changed. These footprints are often not populated. The default connection between pins is a trace on the PCB. To change a cut-trace footprint, the PCB trace must be cut. To return to the original configuration, a header and a jumper must be installed to re-establish the shunt.

NOTE: *Use of the cut-trace header footprints requires a thorough understanding of the MCU and of the EVB hardware. Refer to the MC9S12DP256 Advance Information, Motorola document order number MC9S12DP256TS/D, and to the EVB schematic diagram included on the documentation CD ROM for design information.*

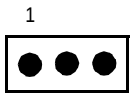
CAUTION: *When cutting a PCB trace to customize a header footprint, use a sharp blade. Be careful to avoid personal injury and not to cut adjacent traces.*

Key to **Table 4-1**: Headers are depicted as viewed from either the component side as shown in **Figure 1-1. EVB Layout and Component Placement** or the solder side as shown in **Figure 1-2. EVB Solder Side View**.

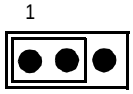
2-pin header with no jumper installed or
2-pin cut-trace header with trace cut

2-pin header with jumper installed

2-pin cut-trace header with default trace intact



3-pin header with no jumper installed



3-pin header with jumper installed on left 2 pins

1–2 **bold** pin numbers indicate factory-default settings
1–2, cut *italics* indicate alternate settings

NOTE: *J21, J26, and J27 are not used.*

Table 4-1. Jumper-Selectable Functions (Sheet 1 of 6)

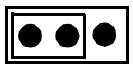
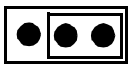
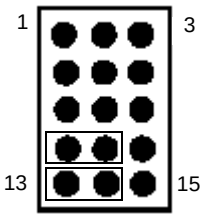
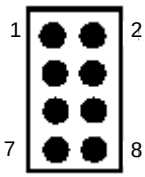
Diagram	Setting	Description
J1 VREGEN		
	**1–2** 2-3	**Use on-chip +2.5V regulator** Use external (on-board) regulator - Voltage should be adjusted with potentiometer R1 and measured at test point TP3 before installation if this option is used.
J2 VRL Voltage Select		
	1–2 **2-3**	A-to-D Voltage Reference Low is pulled high (+5Vdc) **VRL is pulled low (0Vdc)**
J3 CAN2 Control Inputs and Outputs		
		15-pin jumper used to configure control inputs and outputs for CAN2 interface. See schematics for more detail. **(default - jumper pin 10 to 11 and pin 13 to 14)**
J4 CAN2 Physical Interface Connector		
	1 2 3 4 5 6 7 8	GND GND CANH CANH CANL CANL GND GND
J5 VRH Voltage Select		
	**1–2** 2-3	**A-to-D Voltage Reference Low is pulled high (+5Vdc)** VRL is pulled low (0Vdc)
J6 RS232		
		Connects RS232 DCD to DTR **(default – jumper not installed)**

Table 4-1. Jumper-Selectable Functions (Sheet 2 of 6)

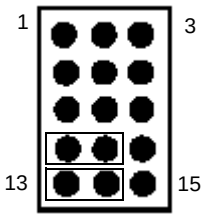
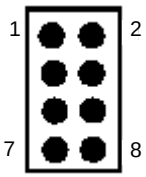
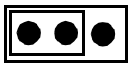
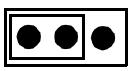
Diagram	Setting	Description
J7 RS232		
		Connects RS232 DSR to DTR **(default – jumper not installed)**
J8 CAN1 Control Inputs and Outputs		
		15-pin jumper used to configure control inputs and outputs for CAN1 interface. See schematics for more detail. **(default - jumper pin 10 to 11 and pin 13 to 14)**
J9 CAN1 Physical Interface Connector		
	1 2 3 4 5 6 7 8	GND GND CANH CANH CANL CANL GND GND
J10 RS232		
		Connects RS232 CTS to RTS **(default – jumper not installed)**
J11 VCC		
	1–2 3–4	Cut trace between 1 and 2 and trace between 3 and 4 to measure current flow on the VDDR, VDDX, and VDDA pins **(default – jumpers not installed)**
J12 RS232		
	**1–2** 2–3	**TXD0 to drive RS232 TX** TXD1 to drive RS232 TX
J13 RS232		
	**1–2** 2–3	**RXD0 to drive RS232 RX** RXD1 to drive RS232 RX

Table 4-1. Jumper-Selectable Functions (Sheet 3 of 6)

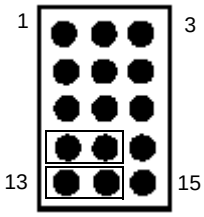
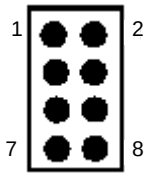
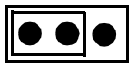
Diagram	Setting	Description
J14 RS232		
	**1–2** 2–3	**Connects LVI circuit to processor reset pin** Connects Resistor-Capacitor network to processor reset pin
J15 CAN0 Control Inputs and Outputs		
		15-pin jumper used to configure control inputs and outputs for CAN0 interface. See schematics for more detail. **(default - jumper pin 10 to 11 and pin 13 to 14)**
J16 CAN0 Physical Interface Connector		
	1 2 3 4 5 6 7 8	GND GND CANH CANH CANL CANL GND GND
J17 VCC		
	**1–2** 2–3	**Use +5Vdc on-board regulator** Use off-board +5Vdc supply
J18 SDI Connector		
	Connector for SDI Connection See schematics for more details 1 2 3 4	 MODA MODB GND PE4

Table 4-1. Jumper-Selectable Functions (Sheet 4 of 6)

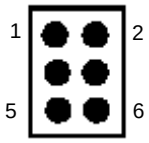
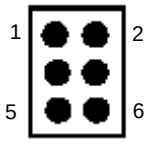
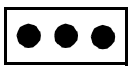
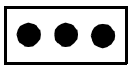
Diagram	Setting	Description
J19 Background Debugger Connector In		
	1 2 3 4 5 6	Connector for Background Debug In BKGD GND NC RESeT(L) NC VCC (+5V)
J20 PAD08		
	1–2 2–3	Connects PAD08 to +5Vdc Connects PAD08 to 0Vdc **(default – jumper not installed)**
J22 Background Debugger Connector Out		
	1 2 3 4 5 6	Connector for Background Debug In PT7 GND NC PT6 NC VCC (+5V)
J23 Power		
	1 2 3	+12Vdc GND +5Vdc (power is normally supplied via J40, +12Vdc, =200ma, center pin positive) **(default – not used)**
J24 PAD00		
	1–2 2–3	Connects PAD00 to +5Vdc Connects PAD00 to 0Vdc **(default – jumper not installed)**
J25 MODC		
	1–2 2–3	Connects MODC to +5Vdc Connects MODC to 0Vdc **(default – jumper not installed)**

Table 4-1. Jumper-Selectable Functions (Sheet 5 of 6)

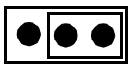
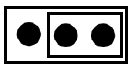
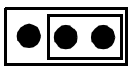
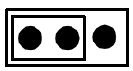
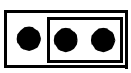
Diagram	Setting	Description
J28 TMOD1		
	1–2 2–3	Connects TMOD1 to +5Vdc Connects TMOD1 to 0Vdc **(default – jumper not installed)**
J29 LED		
		Controls LED operation **(default – jumper installed)**
J30 TMOD2		
	1–2 2–3	Connects TMOD2 to +5Vdc Connects TMOD2 to 0Vdc **(default – jumper not installed)**
J31 External Clock Connector		
		BNC Connector for External Clock **(default – not used)**
J32 PE7		
	1–2 **2–3**	Connects PE7 to +5Vdc **Connects PE7 to 0Vdc**
J33 MODB		
	1–2 **2–3**	Connects MODB to +5Vdc **Connects MODB to 0Vdc**
J34 MODA		
	1–2 **2–3**	Connects MODA to +5Vdc **Connects MODA to 0Vdc**
J35 Oscillator Selector		
	**1–2** 2–3	**Use on-board oscillator** Use external clock (supplied through J31)
J36 TEST		
	1–2 **2–3**	Connects TEST to +5Vdc **Connects TEST to 0Vdc**

Table 4-1. Jumper-Selectable Functions (Sheet 6 of 6)

Diagram	Setting	Description
J37 TERM		
		Provides 50ohm termination in conjunction with external clock (J35) **(default – jumper not installed)**
J38 OSC Enable		
		Install if using a CMOS/TTL clock oscillator (U9). Remove if using a discrete crystal unit. **(default - installed)**
J39 PK7		
	1-2 2-3	Connects PK7 to +5 VDC Connects PK7 to Ground **(default - installed)**

4.3 Power Input Circuitry

The input power connector on the EVB is a 3-pin, lever-actuated connector (J23). Decoupling capacitors filter ripple and noise from the supply voltage.

Fuse (F1) is a thermally activated fuse. An overload will cause the fuse to go to a high impedance state. After power is removed from the board and the fuse cools, the fuse will change to a low impedance state and permit normal operation.

4.4 Terminal Interface

An RS-232C transceiver (U1A or U1B) links the MCU's serial communications interface to the RS-232C DB-9 receptacle, P1.

4.5 Microcontroller

The MC9S12DP256 is one of the first of a family of next generation M68HC12 microcontrollers with both on-chip memory and peripheral functions. The CPU12 is a high-speed, 16-bit processing unit. The programming model and stack frame are identical to those of the standard M68HC11 CPU. The CPU12 instruction set is a proper superset of the M68HC11 instruction set. All M68HC11 instruction mnemonics are accepted by CPU12 assemblers with no changes.

The EVB-resident MC9S12DP256 (U2) has seven modes of operation. These modes are determined at reset by the state of three mode pins — BKGD, MODB, and MODA — as shown in **Table 4-2**.

The EVB is factory-configured for MCU operation in the normal single-chip mode. In this mode of operation, all port pins are available to the user. On-chip FLASH EEPROM is used for program execution, with byte-erasable EEPROM and some RAM available for user code/data. Although other MCU modes are available, the EVB was designed for the single-chip mode of operation. There is no provision for external memory.

For more information on the CPU, refer to the *CPU12 Reference Manual,* Motorola document order number CPU12RM/AD.

Table 4-2. CPU Mode Selection

Input BKGD & bit W25	Input & bit MODB W11	Input &bit MODA W10	Mode Description
0	0	0	Special Single Chip, BDM allowed and ACTIVE. BDM is “allowed” in all other modes but a serial command is required to make BDM “active.”
0	0	1	Emulation Expanded Narrow, BDM allowed
0	1	0	Special Test (Expanded Wide), BDM allowed
0	1	1	Emulation Expanded Wide, BDM allowed
1	**0**	**0**	**Normal Single Chip, BDM allowed**
1	0	1	Normal Expanded Narrow, BDM allowed
1	1	0	Peripheral; BDM allowed but bus operations would cause bus conflicts (must not be used)
1	1	1	Normal Expanded Wide; BDM allowed

There are two basic types of operating modes:

- Normal modes in which some registers and bits are protected against accidental changes.
- Special modes that allow more access to protected control registers and bits for special purposes such as testing.

4.6 Clock Circuitry

The EVB comes with a 16-MHz crystal resonator, U9. The board should be able to accommodate most crystals and oscillators.

Headers J35 and J36 may be used to disconnect the oscillator (U9) or crystal (Y1) from the MCU's on-chip oscillator. An external clock may then be supplied to EXTAL through J31. See the schematics for more detailed information.

4.7 Reset

The reset circuit includes a pullup resistor, reset switch (S1), and a low-voltage inhibit device with a toggle voltage of 4.6 Vdc. This reset circuit drives the MCU's $\overline{\text{RESET}}$ pin directly. Note that header J14 may be used to provide an alternate reset input, provided the cut trace is removed.

4.8 Low-Voltage Inhibit (LVI)

Low-voltage inhibit (LVI) uses an undervoltage sensing device (U5) to automatically drive the MCU's RESET pin low when V_{DD} falls below U5's threshold. This prevents the accidental corruption of EEPROM data if the power-supply voltage should drop below the allowable level.

U5 may be identified by part number MC34164P-5 — 4.5 Vdc.

If operation below U5's threshold (but no less than 2.7 Vdc) is required, one of two methods can be used:

- Replace U5 with a device that has the required threshold voltage.
- Remove the shunt J14 to disconnect U5 from the RESET line. If this is done, an external reset signal should be provided via the center pin of J14 in case the supply voltage falls below the acceptable level.
  Optionally, pins 2 and 3 of J14 can be jumpered to provide an RC type reset function. See the schematic for more detailed information.

4.9 Background Debug Mode (BDM) Interface

The MCU's serial BDM interface can be accessed through two 2-row x 3-pin headers, BDM in (J19) and BDM out (J22). The pin assignments are shown in **Table 4-3**.

The BDM interface may serve in two ways:

- As the "probe" interface through which a host EVB in pod mode controls a target system.
- As the user interface with the EVB. This requires a development tool such as Motorola's serial debug interface. For more information, refer to the *SDI™ Interface User's Manual*, Motorola document order number SDIUM/D.

Table 4-3. BDM Connector J5 Pin Assignments

Pin Number	Description	
	J19 (In)	J22 (Out)
1	BKGD input to MCU	BKGD output from MCU PT7
2	V_{SS}	V_{SS}
3	No connection	No connection
4	$\overline{\text{RESET}}$ input to MCU	$\overline{\text{RESET}}$ output from MCU PT6 (1)
5	No connection	No connection
6	V_{DD}	V_{DD} (1)

(1) Refer to **Table 4-1**.

4.10 Prototype Area

The EVB's prototype area allows construction of custom I/O circuitry that can be connected to the MCU's I/O lines through connectors JP1 through JP8. This area is a grid of holes (approximately 20 by 42) on 1/10-inch (2.54 mm) centers. This spacing accommodates most sockets, headers, and device packages.

Figure 1-1. EVB Layout and Component Placement shows the component-side view of the prototype area. Adjacent ground and V_{CC} (+5V) footprints are provided for wire-wrap pins.

4.11 MCU Connectors

Twelve 2-row x 14-pin header footprints, H1through H4 and JP1 through JP8, surround the MCU and provide access to its I/O and bus lines. They may be populated with wire-wrap pins or strip headers for use as I/O connectors, connection points for instrumentation probes and target hardware, and connections to the prototype area described in **4.10 Prototype Area**. The following figures depict the pin assignments for these headers.

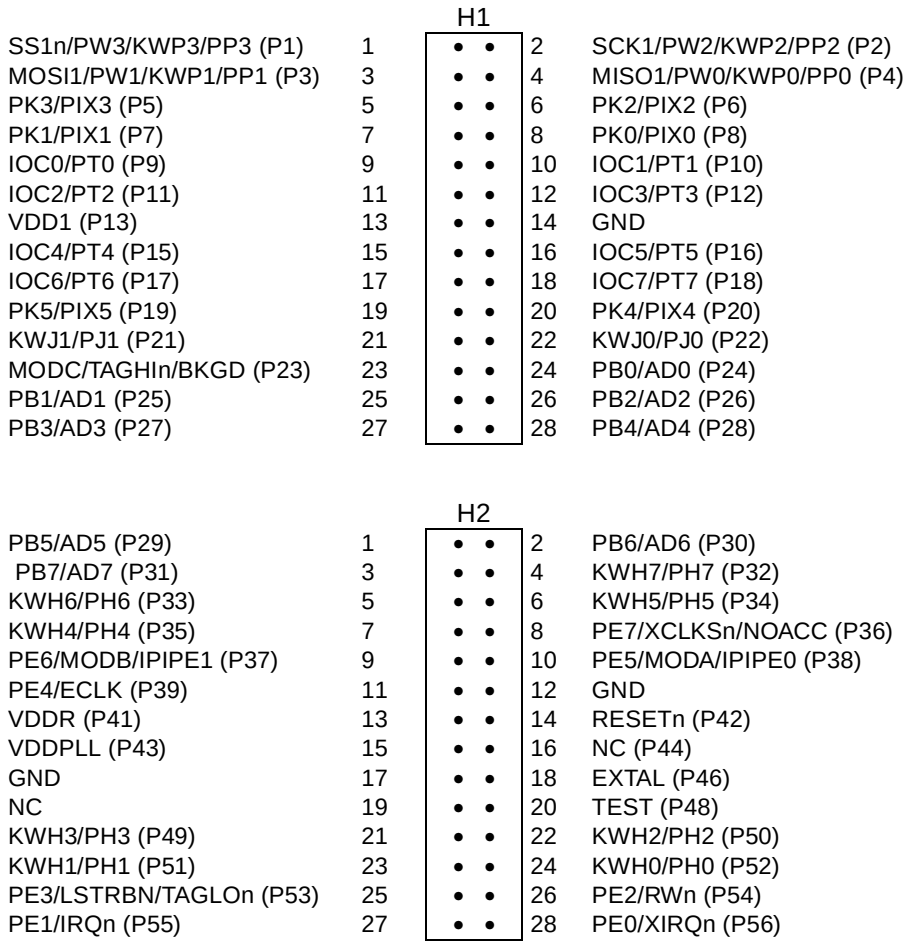


Figure 4-1. MCU I/O Headers H1 and H2

H3

PA0/AD8 (P57)	1	• •	2	PA1/AD9 (P58)
PA2/AD10 (P59)	3	• •	4	PA3/AD11 (P60)
PA4/AD12/TMOD1 (P61)	5	• •	6	PA5/AD13 (P62)
PA6/AD14/TMOD2 (P63)	7	• •	8	PA7/AD15 (P64)
VDD2 (P65)	9	• •	10	GND
PAD00/AN00 (P67)	11	• •	12	PAD08/AN08 (P68)
PAD01/AN01 (P69)	13	• •	14	PAD09/AN09 (P70)
PAD02/AN02 (P71)	15	• •	16	PAD10/AN10 (P72)
PAD03/AN03 (P73)	17	• •	18	PAD11/AN11 (P74)
PAD04/AN04 (P75)	19	• •	20	PAD12/AN12 (P76)
PAD05/AN05 (P77)	21	• •	22	PAD13/AN13 (P78)
PAD06/AN06 (P79)	23	• •	24	PAD14/AN14 (P80)
PAD07/AN07 (P81)	25	• •	26	PAD15/AN15 (P82)
VDDA (P83)	27	• •	28	VRH (P84)

H4

VRL (P85)	1	• •	2	GND
PM7/TXCAN3 (P87)	3	• •	4	PM6/RXCAN3 (P88)
PS0/RXD0 (P89)	5	• •	6	PS1/TXD0 (P90)
PS2/RXD1 (P91)	7	• •	8	PS3/TXD1 (P92)
PS4//SDI/MISO0 (P93)	9	• •	10	PS5/MOSI0 (P94)
PS6/SCK0 (P95)	11	• •	12	PS7/SS0n (P96)
VREGEN (P97)	13	• •	14	PJ7/KWJ7/SCL/TXCAN4 (P98)
PJ6/KWJ6/SDA/RXCAN4 (P99)	15	• •	16	PM5/TXCAN2 (P100)
PM4/RXCAN2 (P101)	17	• •	18	PM3/TXCAN1 (P102)
PM2/RXCAN1 (P103)	19	• •	20	PM1/TXB/TXCAN0 (P104)
PM0/RXB/RXCAN0 (P105)	21	• •	22	GND
VDDX (P107)	23	• •	24	PK7/ECSn (P108)
SCK2/PW7/KWP7/PP7 (P109)	25	• •	26	SS2n/PW6/KWP6/PP6 (P110)
MOSI2/PW5/KWP5/PP5 (P111)	27	• •	28	MISO2/PW4/KWP4/PP4 (P112)

Figure 4-2. MCU I/O Headers H3 and H4

JP1

NC	1	• •	2	SS1n/PW3/KWP3/PP3 (P1)
NC	3	• •	4	SCK1/PW2/KWP2/PP2 (P2)
NC	5	• •	6	MOSI1/PW1/KWP1/PP1 (P3)
NC	7	• •	8	MISO1/PW0/KWP0/PP0 (P4)
NC	9	• •	10	PK3/PIX3 (P5)
NC	11	• •	12	PK2/PIX2 (P6)
NC	13	• •	14	PK1/PIX1 (P7)
NC	15	• •	16	PK0/PIX0 (P8)
NC	17	• •	18	IOC0/PT0 (P9)
NC	19	• •	20	IOC1/PT1 (P10)
NC	21	• •	22	IOC2/PT2 (P11)
NC	23	• •	24	IOC3/PT3 (P12)
VADJ	25	• •	26	VDD1 (P13)
NC	27	• •	28	NC

JP2

NC	1	• •	2	IOC4/PT4 (P15)
NC	3	• •	4	IOC5/PT5 (P16)
PT6	5	• •	6	IOC6/PT6 (P17)
PT7	7	• •	8	IOC7/PT7 (P18)
NC	9	• •	10	PK5/PIX5 (P19)
NC	11	• •	12	PK4/PIX4 (P20)
NC	13	• •	14	KWJ1/PJ1 (P21)
NC	15	• •	16	KWJ0/PJ0 (P22)
BKGD	17	• •	18	MODC/TAGHIn/BKGD (P23)
PB0	19	• •	20	PB0/AD0 (P24)
PB1	21	• •	22	PB1/AD1 (P25)
PB2	23	• •	24	PB2/AD2 (P26)
PB3	25	• •	26	PB3/AD3 (P27)
PB4	27	• •	28	PB4/AD4 (P28)

JP3

PB5	1	• •	2	PB5/AD5 (P29)
PB6	3	• •	4	PB6/AD6 (P30)
PB7	5	• •	6	PB7/AD7 (P31)
PH7	7	• •	8	KWH7/PH7 (P32)
PH6	9	• •	10	KWH6/PH6 (P33)
PH5	11	• •	12	KWH5/PH5 (P34)
PH4	13	• •	14	KWH4/PH4 (P35)
PE7	15	• •	16	PE7/XCLKSn/NOACC (P36)
MODB	17	• •	18	PE6/MODB/IPIPE1 (P37)
MODA	19	• •	20	PE5/MODA/IPIPE0 (P38)
PE4	21	• •	22	PE4/ECLK (P39)
NC	23	• •	24	NC
VDDR	25	• •	26	VDDR (P41)
RESETn	27	• •	28	RESETn (P42)

Figure 4-3. MCU I/O Headers JP1, JP2, and JP3

JP4

VADJ	1	• •	2	VDDPLL (P43)
NC	3	• •	4	NC (P44)
NC	5	• •	6	NC
EXTAL	7	• •	8	EXTAL (P46)
NC	9	• •	10	NC
TEST	11	• •	12	TEST (P48)
PH3	13	• •	14	KWH3/PH3 (P49)
PH2	15	• •	16	KWH2/PH2 (P50)
PH1	17	• •	18	KWH1/PH1 (P51)
PH0	19	• •	20	KWH0/PH0 (P52)
NC	21	• •	22	PE3/LSTRBN/TAGLOn (P53)
NC	23	• •	24	PE2/RWn (P54)
NC	25	• •	26	PE1/IRQn (P55)
NC	27	• •	28	PE0/XIRQn (P56)

JP5

NC	1	• •	2	PA0/AD8(P57)
NC	3	• •	4	PA1/AD9(P58)
NC	5	• •	6	PA2/AD10 (P59)
NC	7	• •	8	PA3/AD11 (P60)
TMOD2	9	• •	10	PA4/AD12/TMOD1 (P61)
NC	11	• •	12	PA5/AD13 (P62)
TMOD1	13	• •	14	PA6/AD14/TMOD2 (P63)
NC	15	• •	16	PA7/AD15 (P64)
VADJ	17	• •	18	VDD2 (P65)
NC	19	• •	20	NC
PAD00	21	• •	22	PAD00/AN00 (P67)
PAD08	23	• •	24	PAD08/AN08 (P68)
NC	25	• •	26	PAD01/AN01 (P69)
NC	27	• •	28	PAD09/AN09 (P70)

JP6

NC	1	• •	2	PAD02/AN02 (P71)
NC	3	• •	4	PAD10/AN10 (P72)
NC	5	• •	6	PAD03/AN03 (P73)
NC	7	• •	8	PAD11/AN11 (P74)
NC	9	• •	10	PAD04/AN04 (P75)
NC	11	• •	12	PAD12/AN12 (P76)
NC	13	• •	14	PAD05/AN05 (P77)
NC	15	• •	16	PAD13/AN13 (P78)
NC	17	• •	18	PAD06/AN06 (P79)
NC	19	• •	20	PAD14/AN14 (P80)
NC	21	• •	22	PAD07/AN07 (P81)
NC	23	• •	24	PAD15/AN15 (P82)
VDDA	25	• •	26	VDDA (P83)
VRH	27	• •	28	VRH (P84)

Figure 4-4. MCU I/O Headers JP4, JP5, and JP6

JP7

VRL	1	• •	2	VRL(P85)
NC	3	• •	4	NC
NC	5	• •	6	PM7/TXCAN3 (P87)
NC	7	• •	8	PM6/RXCAN3 (P88)
RXD0	9	• •	10	PS0/RXD0 (P89)
TXD0	11	• •	12	PS1/TXD0 (P90)
RXD1	13	• •	14	PS2/RXD1 (P91)
TXD1	15	• •	16	PS3/TXD1 (P92)
NC	17	• •	18	PS4//SDI/MISO0 (P93)
NC	19	• •	20	PS5/MOSI0 (P94)
NC	21	• •	22	PS6/SCK0 (P95)
NC	23	• •	24	PS7/SS0n (P96)
VREGEN	25	• •	26	VREGEN (P97)
VCC	27	• •	28	PJ7/KWJ7/SCL/TXCAN4 (P98)

JP8

VCC	1	• •	2	PJ6/KWJ6/SDA/RXCAN4 (P99)
TXCAN2	3	• •	4	PM5/TXCAN2 (P100)
RXCAN2	5	• •	6	PM4/RXCAN2 (P101)
TXCAN1	7	• •	8	PM3/TXCAN1 (P102)
RXCAN1	9	• •	10	PM2/RXCAN1 (P103)
TXCAN0	11	• •	12	PM1/TXB/TXCAN0 (P104)
RXCAN0	13	• •	14	PM0/RXB/RXCAN0 (P105)
NC	15	• •	16	NC
VDDX	17	• •	18	VDDX (P107)
PK7	19	• •	20	PK7/ECSn (P108)
NC	21	• •	22	SCK2/PW7/KWP7/PP7 (P109)
NC	23	• •	24	SS2n/PW6/KWP6/PP6 (P110)
NC	25	• •	26	MOSI2/PW5/KWP5/PP5 (P111)
NC	27	• •	28	MISO2/PW4/KWP4/PP4 (P112)

Figure 4-5. MCU I/O Headers JP7 and JP8

Hardware Reference

Appendix A. S-Record Format

A.1 Overview

The Motorola S-record format was devised to encode programs or data files in a printable format for transport between computer platforms. The format also provides for editing of the S records and monitoring the cross-platform transfer process.

A.2 S-Record Contents

Each S record is a character string composed of several fields which identify:

- Record type
- Record length
- Memory address
- Code/data
- Checksum

Each byte of binary data is encoded in the S record as a 2-character hexadecimal number:

- The first character represents the high-order four bits of the byte.
- The second character represents the low-order four bits of the byte.

The five fields that comprise an S record are shown in **Table A-1**.

Table A-1. S-Record Fields

Type	Record Length	Address	Code/Data	Checksum

The S-record fields are described in **Table A-2**.

Table A-2. S-Record Field Contents

Field	Printable Characters	Contents
Type	2	S-record type — S0, S1, etc.
Record Length	2	Character pair count in the record, excluding the type and record length.
Address	4, 6, or 8	2-, 3-, or 4-byte address at which the data field is to be loaded into memory.
Code/Data	0 – 2n	From 0 to n bytes of executable code, memory loadable data, or descriptive information. For compatibility with teletypewriter, some programs may limit the number of bytes to as few as 28 (56 printable characters in the S record).
Checksum	2	Least significant byte of the one's complement of the sum of the values represented by the pairs of characters making up the record length, address, and the code/data fields.

Each record may be terminated with a CR/LF/NULL. Additionally, an S record may have an initial field to accommodate other data such as line number generated by some time-sharing systems.

Accuracy of transmission is ensured by the record length (byte count) and checksum fields.

A.3 S-Record Types

Eight types of S records have been defined to accommodate the several needs of the encoding, transportation, and decoding functions. The various Motorola upload, download, and other record transportation control programs, as well as cross assemblers, linkers, and other file-creating or debugging programs, utilize only those S records that serve the purpose of the program.

For specific information on which S records are supported by a particular program, the user manual for that program must be consulted.

An S-record format may contain the record types listed in **Table A-3**.

Table A-3. S-Record Types

Type	Description
S0	Header record for each block of S records. The code/data field may contain any descriptive information identifying the following block of S records. The address field is normally 0s.
S1	Record containing code/data and the 2-byte address at which the code/data is to reside
S2 – S8	Ignored by the EVB
S9	Termination record for a block of S1 records. The address field may optionally contain the 2-byte address of the instruction to which control is to be passed. If not specified, the first entry point specification encountered in the object module input will be used. There is no code/data field.

Only one termination record is used for each block of S records. Normally, only one header record is used, although it is possible for multiple header records to occur.

A.4 S Record Creation

S-record format programs may be produced by dump utilities, debuggers, cross assemblers, or cross linkers. Several programs are available for downloading a file in the S-record format from a host system to an 8- or 16-bit microprocessor-based system.

A.5 S-Record Example

A typical S-record format, as printed or displayed, is shown in this example:

Example:

```
S00600004844521B
S1130000285F245F2212226A00042429008237C2A
S11300100002000800082529001853812341001813
S113002041E900084#42234300182342000824A952
S107003000144ED492
S9030000FC
```

In the example, the format consists of:

- An S0 header
- Four S1 code/data records
- An S9 termination record

A.5.1 S0 Header Record

The S0 header record is described in **Table A-4**.

Table A-4. S0 Header Record

Field	S-Record Entry	Description
Type	S0	S-record type S0, indicating a header record
Record Length	06	Hexadecimal 06 (decimal 6), indicating six character pairs (or ASCII bytes) follow
Address	00 00	4-character, 2-byte address field; zeroes
Code/Data	48 44 52	Descriptive information identified these S1 records: ASCII H D R — “HDR”
Checksum	1B	Checksum of S0 record

A.5.2 First S1 Record

The first S1 record is described in **Table A-5**.

Table A-5. S1 Header Record

Field	S-Record Entry			Description	
Type	S1			S-record type S1, indicating a code/data record to be loaded/verified at a 2-byte address	
Record Length	13			Hexadecimal 13 (decimal 19), indicating 19 character pairs, representing 19 bytes of binary data, follow	
Address	0000			4-character, 2-byte address field; hexadecimal address 0000 indicates location where the following data is to be loaded	
Code/Data	Opcode			Instruction	
	28 24 22 22 00 29 08	5F 5F 12 6A 04 00 23	 24 7	BHCC BCC BHI BHI BRSET BHCS BRSET	$0161 $0163 $0118 $0172 0, $04, $012F $010D 4, $23, $018C
Checksum	2A			Checksum of the first S1 record	

The 16 character pairs shown in the code/data field of **Table A-5** are the ASCII bytes of the actual program.

The second and third S1 code/data records each also contain $13 (19T) character pairs and are ended with checksum 13 and 52, respectively. The fourth S code/data record contains 07 character pairs and has a checksum of 92.

A.5.3 S9 Termination Record

The S9 termination record is described in **Table A-6**.

Table A-6. S9 Header Record

Field	S-Record Entry	Description
Type	S9	S-record type S9, indicating a termination record
Record Length	03	Hexadecimal 04, indicating three character pairs (three bytes) follow
Address	00 00	4-character, 2-byte address field; zeroes
Code/Data		There is no code/data in an S9 record.
Checksum	FC	Checksum of S9 record

A.5.4 ASCII Characters

Each printable ASCII character in an S record is encoded in binary. **Table A-5** gives an example of encoding for the S1 record. The binary data is transmitted during a download of an S record from a host system to a 9- or 16-bit microprocessor-based system. For example, the first S1 record in **Table A-5** is sent as shown here.

TYPE				LENGTH				ADDRESS								CODE/DATA								...	CHECKSUM			
S		1		1		3		0		0		0		0		2		8		5		F		...	2		A	
5	3	3	1	3	1	3	3	3	0	3	0	3	0	3	0	3	2	3	8	3	5	4	6	...	3	2	4	1
0101	0011	0011	0001	0011	0001	0011	0011	0011	0000	0011	0000	0011	0000	0011	0000	0011	0010	0011	1000	0011	0101	0100	0110	...	0011	0010	0100	0001

How to reach us:

USA/EUROPE/Locations Not Listed: Motorola Literature Distribution, P.O. Box 5405, Denver, Colorado 80217. 1-303-675-2140 or 1-800-441-2447. Customer Focus Center, 1-800-521-6274

JAPAN: Motorola Japan Ltd.; SPS, Technical Information Center, 3-20-1, Minami-Azabu, Minato-ku, Tokyo 106-8573 Japan. 81-3-3440-8573

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong. 852-26668334

Mfax™, Motorola Fax Back System: RMFAX0@email.sps.mot.com; http://sps.motorola.com/mfax/; TOUCHTONE, 1-602-244-6609; US and Canada ONLY, 1-800-774-1848

HOME PAGE: http://motorola.com/sps/

Mfax is a trademark of Motorola, Inc.

MOTOROLA

MC68EVB912DP256UM/D

Index